4. Fibonacci

Program Name: Fibonacci.java   Input File: fibonacci.dat

The standard textbook example for recursion is the code that computes the terms in the Fibonacci sequence. This is possibly one of the worst examples for recursion. In fact, it should be given as an example of when not to use recursion.

Here is the code as given in most textbooks:

```
public static int fibonacci(int n)
{
    if (n == 0 || n == 1)
      return n;
    else
      return fibonacci(n - 1) + fibonacci(n - 2);
}
```

The n^{th} fibonacci number is defined as the sum of the $(n – 1)^{th}$ fibonacci number and the $(n - 2)^{th}$ fibonacci number. For the base case the 0^{th} fibonacci number is 0 and the 1^{st} fibonacci number is 1.

If we were to expand all of the calls made to the method fibonacci(), you would see that there are a lot of duplicate function calls that end up wasting time and space on the function call stack. For small values of n this is not a problem. For example, for n = 4 there are only 4 duplicate calls. However for bigger values of n the number of duplicate calls escalates.

Given some integer N, determine how many duplicate calls will be made to the fibonacci() method.

Input
The first line of input will contain a single integer T, the number of test cases.
The only line of each test case contains a single integer N.

Output
For each test case, print out the number of duplicated function calls.

Constraints
```
1 <= T <= 10
1 <= N <= 50
```

Example Input File
```
3
4
9
1
```

Example Output to Screen
```
4
99
0
```